Technical Test for AMARIS Interviewees

# Overview:

The objective of this technical test is to evaluate your ability to apply large language models (LLMs) to a real-world task, handle data efficiently in Python, and demonstrate good software design practices. You will build a simple LLM-powered application using provided datasets and handle a data processing challenge as part of this test.

# Scenario:

You are tasked with building a prototype of an "LLM-powered nutrition analysis tool" for Starbucks' menu. The tool should load two datasets (one for drinks, one for food), process the data, and generate natural language summaries of key nutritional insights using the Groq LLM API.

You can use Groq’s free LLM API to interact with the model. For more details, refer to the [Groq Quickstart documentation](https://console.groq.com/docs/quickstart).

# Datasets:

You will be provided with two CSV files:

1. Starbucks Drink Nutrition Data
2. Starbucks Food Nutrition Data

Your task is to process these datasets and extract meaningful insights about the nutrition content of Starbucks menu items.

# Requirements:

## Part 1: Data Loading and Processing

### 1. CSV Data Ingestion:

* Create a Python script that loads the provided Starbucks drink and food CSVs into memory using *`pandas`*.
* Ensure the program can handle common CSV formatting issues (e.g., missing data, inconsistent entries).

### 2. Data Processing:

* Provide basic descriptive statistics for both the drink and food datasets (e.g., total calories, average sugar content, fat-to-protein ratio).
* Compare and contrast key metrics across the two datasets (e.g., average calorie comparison between drinks and food items).
* Generate a visualization (e.g., bar chart, pie chart) comparing key nutritional aspects like calories, sugars, or fats between different menu items.

### 3. Handling Datasets:

* Implement basic handling for datasets by allowing users to filter data based on certain criteria (e.g., show only drinks with caffeine, or food items under 500 calories).

## Part 2: LLM-based Summarization

### 1. Nutritional Summarization:

* Use the Groq LLM API to implement a feature that summarizes nutritional insights from both datasets.
* For example, the LLM could generate a summary about which drink items have the highest sugar content or which food items are most calorie-dense.
* You can find the Groq LLM API documentation and quickstart guide here: [Groq Quickstart Documentation](https://console.groq.com/docs/quickstart).

## Part 3: Code Quality & Documentation

### 1. Modular Code:

* Structure your code into well-organized functions and modules. Ensure separation of concerns between data loading, processing, and summarization.
* Ensure the code is easy to follow and can be extended in the future.

### 2. Documentation:

* Provide clear documentation on how to run the program, including any installation requirements.
* Comment your code sufficiently to explain your logic and reasoning.

## Bonus (Optional):

### 1. LLM-Driven Queries:

* Implement an extra feature where users can input natural language queries about the Starbucks menu data, and the LLM provides an answer (e.g., "What’s the average caffeine content for drinks?").

### 2. Web Interface:

* If you have additional time, create a simple web interface using *`Flask`* or *`Streamlit`* that allows users to upload the CSVs and interact with the tool through their browser.

# Submission Guidelines:

* Zip the entire project, including the code and any necessary files.
* Create a PowerPoint deck as documentation. The deck should cover:
  + An overview of your approach.
  + How to run the application.
  + Key design decisions and trade-offs made during the implementation.
  + Screenshots or diagrams where applicable.
* Email the zipped project and the PowerPoint deck back.

# Evaluation Criteria:

* **Functionality:** Does the tool meet the requirements? Are the datasets processed correctly? Is the summarization meaningful?
* **Code Quality:** Is the code clean, modular, and well-documented?
* **Efficiency:** Does the tool handle the datasets efficiently?
* **Creativity:** Are there any additional features or unique approaches that demonstrate creativity in problem-solving?